

1. Classes, Objects and the UML Class Diagram

M2104 - Object Oriented Design & Modelling

2021 Edition

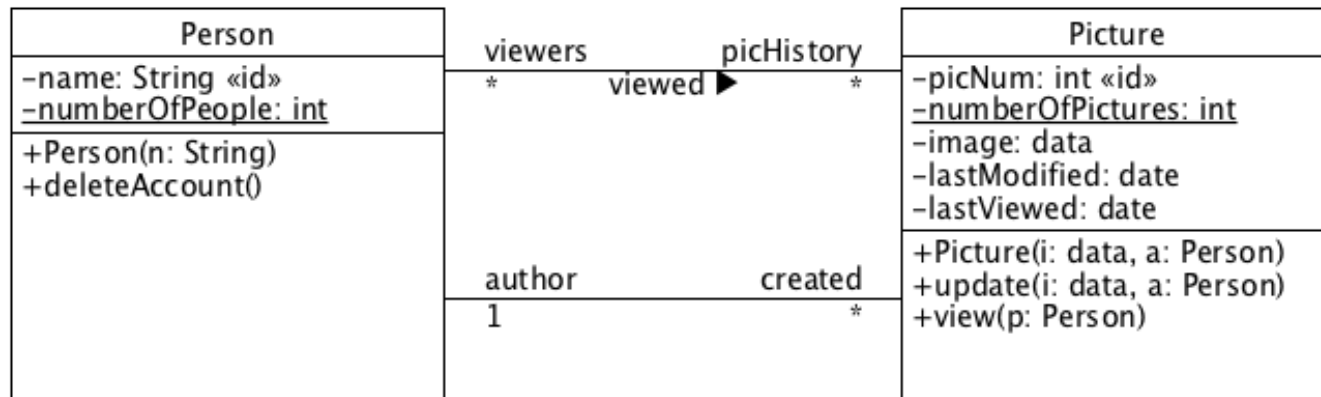
Design & Modelling

- ❖ **Design** is to prepare software development and to clarify the project expectation. Design is **synthesis**, resulting in software **specifications**, it is a **creative** activity.
- ❖ **Modelling** is the making of a **model** for some phenomenon. It is the **abstraction** of part of the observed / imagined world. Modelling is **analysis**, resulting in process **specifications**, it is a **summary**.
- ❖ Using both **design and modelling** in an **object-oriented paradigm**, software developers create **full specifications for software** encompassing **needs and processes**.
- ❖ The goal of the specification documents is to serve as **reference** for the development and to be **communicated** between teams.

Classes & Objects

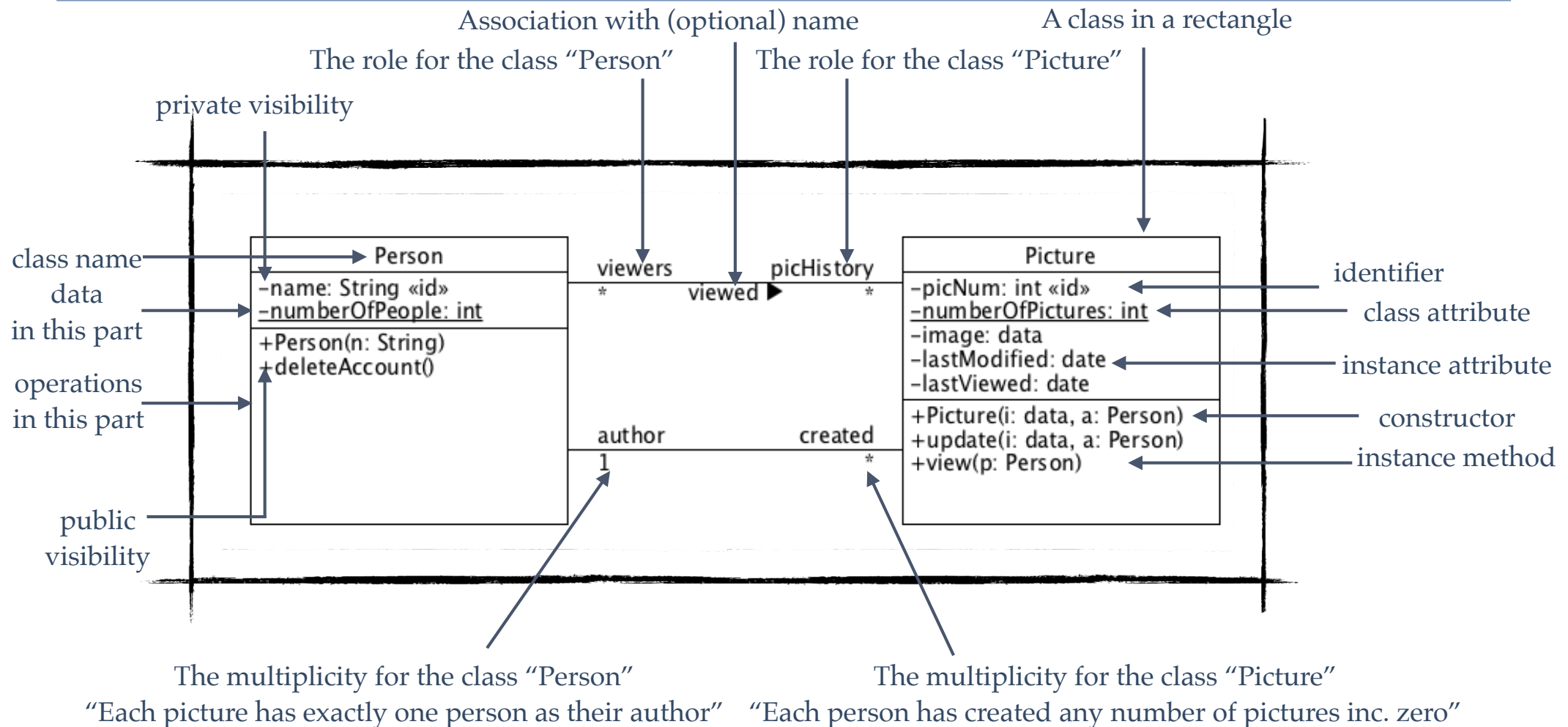
- ❖ **Objects** are class **instances**, **classes** are specifications for **objects**.
- ❖ Objects are **service providers** and also contain **relevant data**.
- ❖ The services provided by objects are operations, **methods**, the data necessary for objects are stored in **attributes**.
- ❖ Classes can also provide **entity-independent services and data** in the form of **class methods** and **class attributes** (not linked to any specific object).
- ❖ **Associations between classes** are references between objects of those classes (an object of one contain a reference to object(s) of the other).
- ❖ Associated objects and objects **using** other classes form a diagram of **relations** evidenced in the **UML class diagram**.

The UML Class Diagram Example



The UML Class Diagram

Comments



The UML Class Diagram Summary

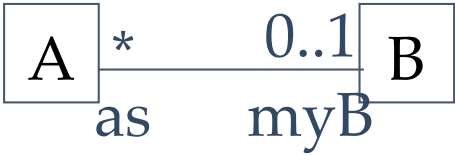

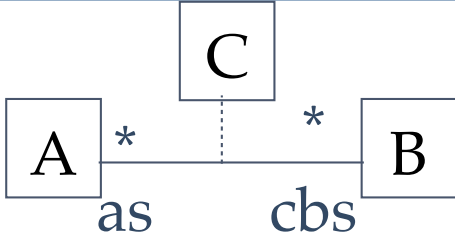
	UML	Meaning	Java
Class	Rectangle box	A blueprint for objects	<code>class</code>
Attribute	visibility name : type	Data	<code>visibility type name;</code>
Method	visibility name(argument list) : return type	Operation	<code>visibility type name (argument list) {}</code>
Visibility	+ / - / ~ / #	Provided to others / Only necessary for the class / ...	<code>public / private / / protected</code>
For class / instance	Underline for class	Class attributes and methods are not linked to specific objects	<code>static</code>
Associations	Straight line with roles and multiplicities	The fact that objects contain references to other objects	Attributes of the respective classes

Associations & Containers

- ❖ Multiplicity 1 or 0..1 : reference to an object that can be null.
- ❖ Pluralities : containers such as *collections* including:
 - ❖ Standard Java arrays / lists (*ArrayList*)
 - ❖ *Sets* (*TreeSet*, *HashSet*)
 - ❖ Associative mappings : *Maps* (*TreeMap*, *HashMap*)
- ❖ Association classes can represent mappings or other relations

Associations & Containers

Examples

Singular multiplicity		<pre>class A { B myB; }</pre>
Plural multiplicity		<pre>class A { List myBs; }</pre>
Association class		<pre>class A { Map<B, C> cbs; }</pre>